PENGGUNAAN TEKNIK *FEATURE WEIGHTING* UNTUK PEMBERSIHAN *NOISE* PADA HALAMAN SITUS BERITA BERBAHASA INDONESIA

Rahmat Firdaus[1] ZK. Abdurahman Baizal[2] Yanuar Firdaus A. W[1]

[1]Program Studi Teknik Informatika , Institut Teknologi Telkom, Bandung
[2]Program Studi Ilmu Komputasi , Institut Teknologi Telkom, Bandung
Email: mamatdaus@gmail.com, zka@ittelkom.ac.id, yfa@ittelkom.ac.id

ABSTRACT *A web page usually consists of information in every page blocks displayed. In some cases, news content displayed in a news website are not entirely relevant or are unrelated to the main content such as navigation panel, copyright, user guide, links, news summary, various advertisement etc. Information blocks irrelevant to the main content is known as web pages noise. This research applies feature weighting technique to improve classification results by detecting a noise in pages of a website. Using feature weighting technique the web is first modelled with Document Object Model (DOM) tree and Compressed Structure Tree(CST) to obtain the general structure and compare the information blocks in a website.Information obtained is used to measure and evaluate the importance level of each node created by Compressed Structureed Tree(CST). Based on the tree created and the importance level of each node, this method assign weights on each individual word (feature) in each content block. The weights will be used in web mining process.*

Keywords: *CST, DOM, noise detection, noise elimination, web mining.*

ABSTRAK *Suatu halaman situs berita banyak mengandung isi informasi dalam tiap-tiap blok halaman yang ditampilkan. Kadangkala isi berita yang ditampilkan pada halaman berita di suatu situs tidak memberikan informasi yang sepenuhnya relevan atau tidak berhubungan dengan isi utama misalnya, panel navigasi, copyright, user guide, links, sinopsis suatu berita, berbagai macam iklan dan lain-lain. Blok-blok informasi yang tidak relevan dengan isi utama tersebut dikenal sebagai web pages noise yang menurunkan hasil kinerja klasifikasi. Dalam penelitian ini penggunaan teknik feature weighting untuk meningkatkan performansi hasil klasifikasi dengan mendeteksi noise yang ada pada halaman website. Dengan teknik feature weighting ini suatu halaman web terlebih dahulu akan dimodelkan dengan pohon struktur Document Object Model (DOM) tree dan Compressed Structure Tree(CST) untuk memperoleh struktur umum dan membandingkan blok-blok informasi dalam suatu website. Informasi yang didapatkan digunakan untuk melakukan pengukuran dan mengevaluasi tingkat kepentingan dari masing-masing node yang terbentuk dari Compressed Structureed Tree(CST). Berdasarkan tree yang terbentuk dan tingkat kepentingan dari nilai node yang didapatkan, metoda ini memberikan bobot pada masing-masing individual word (feature) pada masing-masing blok isi. Hasil pembobotan (weight) digunakan dalam proses web mining. Teknik ini berhasil membersihkan blok noise dan meningkatkan hasil klasifikasi berdasarkan uji coba pada 4 dataset.*

Kata Kunci: *CST, DOM, deteksi noise, eliminasi noise, web mining.*

Semakin meluasnya pengunaan Internet telah memacu pertumbuhan dan pertukaran informasi yang sangat pesat dibandingkan era sebelumnya. Sebagai akibatnya, jumlah informasi terus meningkat secara eksponensial. Sumber informasi tidak hanya dalam bentuk dokumen atau teks tetapi di internet informasi juga dihasilkan, dikoleksi dan dikonsumsi dalam jumlah yang tidak kalah besar oleh berbagai institusi dan perusahaan.

Sumber informasi yang ditampilkan oleh suatu situs di internet merupakan sumber informasi yang sangat beraneka ragam. Ada berbagai macam jenis situs dengan berbagai macam informasi yang dibawa. Secara umum tidak seperti kebanyakan data atau teks konvensional lainnya suatu halaman web selain mengandung konten utama juga mengandung banyak blok informasi yang tidak berhubungan dengan konten utama misalnya, panel navigasi, *copyright*, *user guide*, kumpulan *link*, sinopsis suatu berita, berbagai macam iklan dan lain-lain. Secara keseluruhan adanya hal tersebut secara tidak langsung bertujuan untuk memudahkan user dalam mengakses web dan atau untuk tujuan bisnis tertentu. Dalam suatu website terdapat satu elemen yang meng-andung satu topik tunggal pada suatu blok tertentu [1, 2, 3]. Blok-blok informasi yang tidak relevan dengan konten utama pada suatu halaman web disebut sebagai *web pages noise*. Adanya informasi yang tidak relevan (*noise*) dalam suatu sumber data (*web pages*) dapat menjadi permasalahan yang serius dalam proses data mining misalnya dalam proses klasifikasi atau klasterisasi.

Dalam penelitian ini akan digunakan diusulkan penggunaan teknik *feature weighting* untuk mendeteksi dan mengurangi *noise* yang ada dalam suatu halaman web. Dengan metoda ini suatu halaman web akan dimodelkan dengan struktur pohon *Dokumen Object Model* (DOM) *tree* dan *Compressed Structure Tree* (CST) untuk mendapatkan struktur umum dan blok informasi dalam suatu website.

Dataset yang diambil adalah *html pages* secara *offline* yang berupa berita berbahasa Indonesia. Setiap halaman web berita sudah mempunyai layout standar dalam arti tiap halaman yang satu dengan yang lain memiliki kesesuaian bentuk umum, misalnya dibangun dengan CMS (*Content Management System*).

WEB PAGES NOISE

Secara umum suatu halaman web berisi sejumlah besar informasi yang bukan merupakan bagian utama dari halaman tersebut, contohnya iklan, pesan hak cipta, *navigational bar*, dan sebagainya. Informasi yang tidak relevan tersebut disebut dengan *web pages noise* [4]. Berdasarkan granulitasnya, *noise* pada halaman web dibagi menjadi dua macam, yaitu *Global Noise* dan *Local Noise*.

Global noise adalah *noise* yang mengacu pada objek yang berulang *redundant* dengan granulitas yang besar [5]. Misalnya *mirror site* yaitu, halaman web yang terduplikasi dan versi terdahulu dari halaman web. Sedangkan *local noise* adalah *noise* yang mengacu pada item yang tidak relevan dalam sebuah halaman web [5]. *Local noise* pada umumnya tidak mengacu pada isi utama dari halaman web tersebut.

Terdapat empat kategori *noise* pada halaman web [5] yaitu 1) *fix description noise*, 2) *service noise*, 3) *navigational guidance* dan 4) *advertisement*. Fix description noise adalah *noise* yang bersifat tetap misalnya logo situs, dekorasi image, pesan hak cipta, *private statement* dan lain-lain. Service noise adalah informasi yang bersifat pelayanan yang tidak relevan, misalnya perkiraan cuaca, pasar saham dan lain-lain. Pemandu direktori atau *link* adalah *navigational guidance*. Advertisement adalah *noise* yang bersifat mengiklankan berbagai macam produk.

DATA *PREPROCESSING*

Data *preprocessing* dibutuhkan untuk mengurangi dimensi dari data yang sangat besar, data yang mengandung *noise*, duplikasi data, data yang tidak lengkap dan lain-lain. Keuntungan dari *Dimensionality Reduction* adalah mengura-ngi ciri-ciri yang tidak berpengaruh (*noise*), sehingga keuntungan lain dari hasilnya dapat ditingkatkan [6]. Jumlah waktu dan memori yang dibutuhkan pada algoritma akan berkurang seiring dengan pengurangan dimensi yang dilakukan.

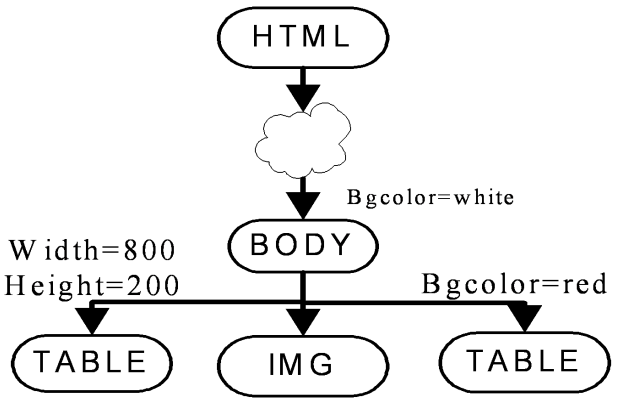
Beberapa cara yang dapat dilakukan pada proses *preprocessing* khususnya dalam data teks adalah dengan menghilangkan *noise* yang terdapat dalam data, melakukan proses segmentasi data, penghilangan *stopword*, dan *indexing*.

TEMPLATE DAN *PAGELETES*

Sebuah website modern mengandung banyak elemen untuk memudahkan user mengakses setiap halaman. Tiap e-lemen terletak pada tiap blok tertentu dalam suatu halaman web. Sebuah *pagelet* adalah sebuah *logical region* yang hanya mengandung satu fungsionalitas atau topik tunggal [1, 2, 3, 7]. Suatu *page* dapat dibagi menjadi beberapa *pagelets* yang sesuai dengan topik dan fungsionalitas yang berbeda. Sebuah *template* tertentu digunakan agar informasi yang diberikan mudah untuk dilihat dan diakses. Sebuah *template* adalah *master* HTML *shell* yang dipersiapkan dan digunakan sebagai basis untuk menyusun dan mengubah halaman web yang baru [1, 7].

DOKUMEN OBJEK MODEL

DOM (*Document Object Model*) merupakan bentuk representasi dari sebuah halaman web. DOM berupa halaman yang mengandung format HTML (*Hypertext Markup Language*) atau XML (*Extended Markup Language*). Suatu halaman web yang mempunyai format HTML akan diubah kedalam bentuk DOM *tree*, sehingga memiliki bentuk umum standar yang dapat digunakan untuk memanipulasi dan mengakses dokumen HTML. HTML cocok untuk digambarkan dalam DOM *tree* karena segala sesuatu dalam dokumen HTML adalah sebuah *node* [8].



Gambar 1: Dokumen Objek Model *tree*

Berdasarkan w3school [8] DOM memiliki beberapa definisi diantaranya adalah: 1) semua dokumen HTML didefinisikan sebagai dokumen *node*, 2) setiap *tag* HTML adalah sebuah elemen *node*, 3) teks yang berada dalam elemen HTML adalah teks *node*, 4) setiap atribut HTML adalah atribut *node*, 4) suatu komentar dalam dokumen HTML merupakan komentar *node*.

Setiap *node* memiliki hubungan secara hirarki satu sama lain. Semua *node* dalam dokumen HTML dibentuk dalam sebuah dokumen *tree*. Gambar 1 menunjukkan contoh pohon dari dokumen HTML sebagai berikut :

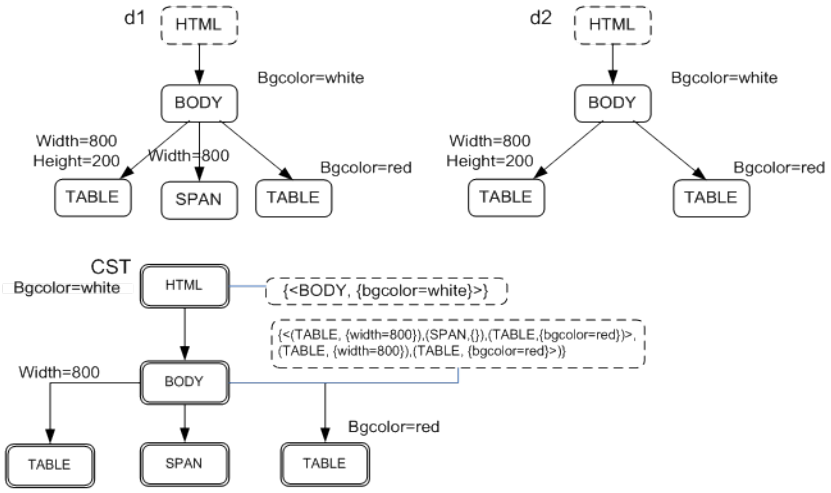
$$
\begin{aligned}
&< BODY bgcolor = WHITE >\\
&< TABLE > width = 800 height = 200 >\\
&\ldots\\
&< /TABLE >\\
&< IMG src = "image.gif" width = 800 >
\end{aligned}
$$

Dalam DOM *tree*, *presentation style* dari sebuah *tag node* T didefinisikan sebagai S_T , dimana S_T adalah serangkaian dari <r_1, r_2, $\ldots r_n$> dan r_i adalah pasangan antara (*tag*, Attr) yang menggambarkan child *tag node* ke-i dari T [4].

Tag adalah nama dari *tag* HTML, misalnya : "TABLE" dan "IMG" sementara Attr adalah sekumpulan dari display atribut dari suatu *tag*, misalnya: bgcolor=RED, width=100 dan lain-lain.

Keterangan "n" adalah panjang dari style yg ada. Contoh pada gambar 1, *presentation style* dari *tag node* BODY adalah <(TABLE, width=800, height=200), (IMG, width= 800), (TABLE, bgcolor=red)>.

Dua buah *presentation style* S_a:<r_{a1},r_{a2},$\ldots$,r_{am}> dan S_b:<r_{b1},r_{b2},$\ldots$,r_{bn}> dikatakan sama S_a=S_b jika dan hanya jika memiliki kondisi m = n dan $r_{ai} \cdot Tag = r_{bi} \cdot Tag$ dan $r_{a1} \cdot Attr$=$r_{bi} \cdot Attr$, dengan i=1,2,..m



Gambar 2: DOM *tree* dan CST

COMPRESSED STRUCTURE TREE (CST)

CST merupakan sebuah *tree* yang merupakan hasil gabungan dari beberapa DOM *tree* untuk memperlihatkan struktur umum dari sebuah website. CST dibuat untuk menanggulangi kekurangan dari DOM *tree* yang hanya merepresentasikan struktur untuk satu halaman web saja dan tidak bisa merepresentasikan struktur umum dari sekumpulan halaman web.

Sebuah elemen *node* E pada CST, menggambarkan sekumpulan gabungan dari *tag node* yang ada pada DOM *tree*. CST memiliki lima komponen yang disimbolkan oleh *tag*, Attr, TAGs, STYLEs, CHILDs. *Tag* adalah nama dari *tag* HTML sementara Attr adalah sekumpulan display atribut dari *tag*. TAGs adalah sekumpulan actual *tag* nodes dalam DOM *tree* yang telah dicompress atau gabung di E. STYLEs adalah sekumpulan *presentation style* yang telah digabung di E. CHILDs adalah sekumpulan pointer yg menunjuk child elemen *node* E di CST.

Dari contoh gambar 2 dapat dilihat bahwa *Compressed Structure Tree* (CST) merupakan hasil gabungan dari d1 dan d2. Dapat kita amati bahwa semua *tag node* pada *tree* d1 dan d2 memiliki kecocokan terkecuali *tag node* SPAN, sehingga sebuah elemen *node* pada CST sebenarnya merupakan sekumpulan dari *tag node* yang terbentuk dari berbagai macam DOM *tree*.

CST dibangun dari sejumlah halaman web suatu website dengan menggabungkan DOM *tree* secara *top-down* terdiri dari beberapa langkah [5]. Langkah pertama semua *root tag nodes* dari DOM *tree* di gabung kedalam bentuk dasar (top most) elemen *node* dari CST. Disini *Tag*=root atau *Tag*=HTML, Attr= dan TAGs menjadi kumpulan dari semua *root tag nodes* dalam DOM *tree* dari halaman situs. Kemudian pada langkah kedua dilakukan proses penghitungan STYLEs dari elemen *node* E dilanjutkan dari langkah sebelumnya. E.STYLEs adalah kumpulan dari *presentation style* dari semua *tag* nodes dalam DOM *tree* tercakup oleh E. Pada langkah ketiga semua child *tag* nodes yang bersesuaian dalam E.TAGs dengan *presentation style* yang sama di gabung, dengan bentuk *initial child element nodes* dari E. Untuk tiap pasangan dari *initial child element node* E1 dan E2, jika secara berturut-turut *Tags* dan *Attr* memiliki kesamaan, maka isi teks dari E1 dan E2 akan dibandingkan apakah memiliki kesamaan dan dapat di gabung. Jika tidak ada lagi *child element node* yang dibuat pada langkah ketiga, maka proses akan dihentikan, jika tidak untuk setiap *child element node* dari langkah ke tiga akan berlanjut ke langkah ke dua.

CST yang telah dibuat, akan digunakan untuk menghitung bobot dari masing-masing kata di masing-masing blok dari halaman web.

PEMBOBOTAN

Pada bagian ini akan ditampilkan langkah-langkah pembobotan.

Langkah pertama adalah pembobotan TF. Pembobotan TF (*Term Frequency*) menghitung jumlah kemunculan term dalam dokumen. Semakin banyak kemunculan term, maka akan semakin besar pula bobot TF term tersebut

$$TF(d,t) = f(d,t) \tag{1}$$

$f(d,t)$: frekuensi term t dalam dokumen d

Suatu term menjadi kurang diskriminatif jika term tersebut muncul di banyak dokumen. Oleh karena itu pembobotan IDF digunakan dalam menghitung bobot term.

$$IDF(t) = \log\left(\frac{N}{n_t}\right) \tag{2}$$

N : jumlah dokumen
n_t : jumlah dokumen yang mengandung term t

Pada pembobotan TF-IDF term dihitung dengan:

$$w_{kj} = TF(j,k) \times IDF(k) \tag{3}$$

w_{kj} : bobot term ke-k dalam dokumen j

Teknik pembobotan dengan *feature weighting* akan melewati beberapa langkah perhitungan dimulai dari menghitung nodeImp, pathImp dan *weighting Feature*.

Definisi: Untuk sebuah internal elemen *node* E di CST, misalkan l = |E.Styles| dan m= | E.Tags|. Pentingnya E, ditandai oleh Nodeimp(E) yang dirumuskan:

$$NodeImp(E) = \begin{cases} 1 & if m = 1 \\ \sum_{i=1}^{l} p_i \log_m p_i & if m > 1 \end{cases} \quad (4)$$

dengan p_i adalah kemungkinan bahwa suatu *tag node* di E.TAGs menggunakan presentasi style ke-i.

Kemudian *leaf node* diperlakukan berbeda dari sebuah internal *node* karena mengandung kata aktual tanpa *tag* apapun.

$$NodeImp(E) = 1 - \frac{\sum_{i=1}^{N}(H_E(a_i))}{N} = \frac{\sum_{i=1}^{N}(1 - H_E(a_i))}{N} \quad (5)$$

dengan a_i adalah *feature* dari isi S di E dan HE = (a_i) adalah informasi entropy dari a_i di dalam E, yang mana H_E dirumuskan sbb:

$$H_E(a_i) = \begin{cases} 1 & if m = 1 \\ \sum_{i=1}^{m} p_i \log_m p_i & if m > 1 \end{cases} \quad (6)$$

dengan m= | E.Tags|, dan p_{ij} adalah kemungkinan dimana a_i muncul dalam *tag node* ke-j dari E.TAGs

Nodeimp(E) hanya mengevaluasi tingkat kepentingan secara lokal di E. Agar dapat memberikan pembobotan *feature* yang ada pada *leaf node* dari CST, kita menggunakan alur *cumulative importance* yaitu tingkat kepentingan secara keseluruhan dari root ke *node* yang mengandung *feature*. *Cumulatif importance* dari E disebut *path importance*, yang disimbolkan oleh PathImp(E). *Path importance* mengukur pentingnya struktur dari root ke E di CST.

Definisi: Untuk tiap elemen *node* E di CST, *path importance* dari E, disimbolkan sebagai PathImp(E) yang ditunjukkan pada persamaan 7 :

$$PathImp(E) = 1 - \prod_{E_i \cdot Ancestor(E) \cup E} (1 - NodeImp(E_i)) \quad (7)$$

dengan E_i adalah *ancestor* (nenek moyang) dari E atau E sendiri dalam CST.

Kebijakan weighting mempertimbangkan dua hal yaitu struktur dan konteks isi *feature*. Bobot dari tiap *feature* dalam blok tertentu (atau di bawah *tag* tertentu) dari halaman Web dihitung sebagai berikut:

Definisi: Untuk sebuah *feature* a_i didalam suatu *leaf* elemen *node* E dari CST, bobot a_i dibawah *tag node* T_j dari DOM *tree* dari suatu halaman Web (Tj ∈, E.TAGs), ditandai oleh :$W_E(a_i, T_j)$, digambarkan oleh :

$$W_E(a_i, T_j) = PathImp(E) \times (1 - H_E(a_i)) \times f_{ij} \quad (8)$$

PARAMETER EVALUASI DAN PERANCANGAN SISTEM

Cara yang umum dipakai untuk mengukur performansi *classifier* adalah dengan menghitung precision dan recall kemudian menghitung *F-measure* seperti yang ditunjukkan pada persamaan 9 - 11.

$$precision = \frac{Numberofcategoriesfoundthatarecorrect}{Totalnumberofcategoriesfound} \quad (9)$$

$$recall = \frac{Numberofcategoriesfoundthatarecorrect}{Totalnumberoftruecategories} \quad (10)$$

$$F_1measure = \frac{2 \times precision \times recall}{precision + recall} \quad (11)$$

Secara garis besar, sistem yang dibangun dapat ditunjukkan pada gambar 3. Seperti yang ditunjukkan pada gambar 3 langkah pertama dalam sistem ini adalah *crawling*. Situs web dari Internet akan di-*crawling* atau dilakukan proses *download* dan di-copy kedalam *storage device* menggunakan tools Teleport. Proses *crawling* dilakukan pada kedalaman 3 sampai 5 level dari situs web yang telah ditentukan. Proses *crawling* dilakukan per situs web.

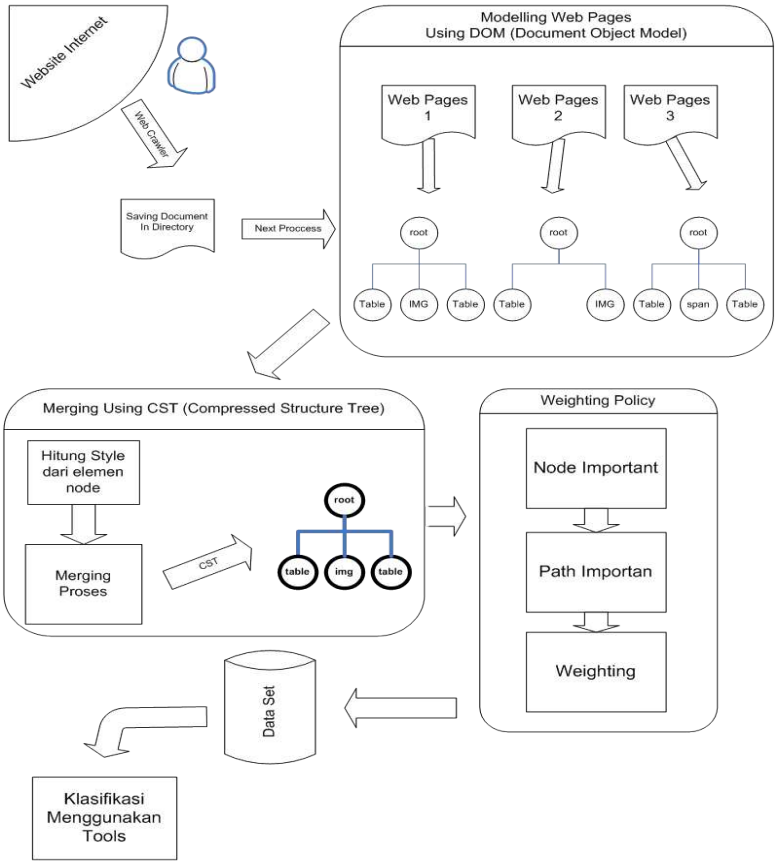
Langkah selanjutnya adalah *Modelling web pages*, yaitu proses untuk menemukan struktur umum dari setiap halaman web dari situs yang sama. Proses pemodelan dilakukan dengan membangun pohon (DOM *tree*) dari setiap halaman website dengan mengacu pada setiap *tag-tag* HTML yang ditemukan pada tiap halaman web. Pada proses ini dilakukan parsing *tag-tag* HTML dan teks yang diapit oleh tiap *tag* HTML untuk kemudian akan dilakukan proses penghilangan *stopword* yaitu kata yang dianggap tidak memiliki makna penting atau sering muncul dan *case-folding* yaitu proses penghilangan tanda baca angka yang terkandung pada teks dokumen.

Setelah pemodelan langkah berikutnya adalah *Merging Dokumen*, yaitu proses penggabungan dari tiap model *tree* yang telah terbentuk dari proses sebelumnya. Proses penggabungan akan melihat dan menghitung tiap style yang dimiliki oleh tiap dokumen. Proses ini akan menghasilkan pohon *Compressed Structure Tree* (CST) yang akan dijadikan acuan dalam proses pembobotan selanjutnya. Kemudian dilakukan *Weighting Policy* yang merupakan proses perhitungan bobot tiap *feature* berdasarkan CST yang terbentuk dari proses sebelumnya. Perhitungan bobot dilakukan dengan menghitung *inner node importance*, *leaf node importance*, *information entropy of feature*, *path importance* dan *weight of feature*. Tiap *feature* yang telah diberikan bobot akan disimpan kedalam basis data untuk kemudian akan diekspor kedalam *file record* yang akan digunakan sebagai masukan dalam proses klasifikasi.

UJI COBA

Deskripsi Sistem

Secara umum sistem yang akan dibangun adalah memberikan pembobotan tiap kata (*feature*) pada seluruh halaman website yang ada dan menemukan informasi yang sesuai dan informatif pada suatu halaman web dengan melakukan sampling pada beberapa halaman web yang ada. Kemudian analisis berdasarkan bobot yang ada (*feature*



Gambar 3: Urutan proses deteksi dan flagging website

weight) yang didapat dari *Compressed Structure Tree* (CST). Hasil dari keluaran bobot tersebut akan digunakan dalam proses data mining yaitu proses klasifikasi menggunakan *classifier* yang terdapat pada tools WEKA.

Data Input dan Output

Secara umum masukan untuk perangkat lunak ini adalah data mentah yang berupa file berformat HTML yang berasal dari situs web berita berbahasa Indonesia. Pengguna akan menginputkan jumlah halaman website yang akan dilakukan proses *flagging* dengan sebelumnya dilakukan proses *crawling* dari sumber berita secara online dengan menggunakan *tools teleport*. Sedangkan yang menjadi output pada perangkat lunak tersebut adalah data berformat csv dimana setiap *feature*-nya telah diberikan bobot sebagai prediktor untuk menentukan kelas. Kemudian akan dilakukan proses pengklasifikasian untuk mengecek apakah hasil akurasi meningkat atau tidak.

Skenario Pengujian

Pengujian dilakukan untuk melihat seberapa baik proses deteksi dan pembobotan yang dilakukan oleh sistem dengan pendekatan text mining. Proses evaluasi dari pengujian dilihat dari nilai *F-measure* dan akurasi yaitu ketepatan klasifikasi data. Proses pengujian dilakukan sebanyak lima kali percobaan untuk setiap jenis pengujian dengan mengubah-ubah nilai random pada WEKA. Proses training menggunakan empat dataset berbeda dengan jumlah total 400 dokumen, serta dilakukan proses *feature selection* dengan melakukan perangkingan bobot dan *treshold* berupa jumlah kata yang dihasilkan yaitu 100, 500, 1000 dan 5000. Proses pengujian menggunakan data sebanyak 20% dari data training yang diambil secara random dari 100 data.

Dataset yang akan digunakan berjumlah 100 data untuk masing-masing situs penyedia. Terdapat beberapa kategori pelabelan dari dataset yang diambil yang dijelaskan dalam tabel 1.

Pengujian Data Clean

Dataset yang telah dibersihkan dari *noise* atau data *clean* dari keempat situs berita didapat dengan dua cara yaitu menggunakan sistem yang dibangun yang dibuat berdasarkan teknik *feature weighting* dan yang kedua data dibersihkan secara manual dengan *human analisis*. Data *clean* yang dibersihkan secara manual yang sudah terbebas dari *noise* dilakukan berdasarkan pengamatan secara umum dari situs yang telah disediakan.

Pengujian Data Yang Mengandung *noise*

Dataset sebelum dideteksi atau masih mengandung *noise* dilakukan oleh sistem dengan terlebih dulu memisahkan teks dan *tag-tag* HTML yang dikandung oleh file yang berformat HTML. Setelah itu data akan disimpan kedalam

Tabel 1: Dataset dan kategori dari empat situs berbeda

No	Dataset	Kategori
1	Kompas	Bisnis keuangan, Kesehatan, Olahraga, Property, Travel
2	Pikiran	Rakyat Ekonomi, Nasional, Olahraga
3	Liputan6	Hukrim, Olahraga, Politik
4	Jawapos	Ekonomi Bisnis, Politik, Sport

file berformat txt untuk kemudian dilakukan proses klasifikasi dengan *classifier* yang ada pada tools WEKA.

Pengujian data sebelum dihilangkan noisenya dilakukan dengan maksud untuk mengukur seberapa baik atau buruk proses deteksi *noise* yang dilakukan oleh sistem. Evalusi akan memperhatikan nilai akurasi dan *F-measure* yang dihasilkan dari proses klasifikasi.

Pengujian F-Measure

Hasil distribusi *F-measure* dari dataset kompas dapat dilihat pada gambar 4. Persebaran nilai tertinggi secara umum terdapat pada kategori olahraga, kesehatan, dan bisnis keuangan yang berkisar antara 0,900 sampai 1,000. Ketepatan dokumen diklasifikasikan berdasarkan kelasnya berada pada dataset yang telah dihilangkan noisenya dengan human analisis.

Untuk dataset yang dibersihkan dengan sistem, persebaran *F-measure* terbaik berada pada pemilihan *feature* sebanyak 5000, yaitu pada kategori olahraga, bisnis keuangan dan kesehatan. Semakin banyak *feature* yang dipilih untuk dataset yang dibersihkan dengan sistem maka akan semakin tepat pengklasifikasian dokumen berdasarkan kelasnya. Hal ini dapat dijelaskan bahwa proses pembobotan dengan *feature weighting* akan membobotkan setiap *feature* pada keseluruhan dokumen, sehingga semakin banyak *feature* relevan yang dipilih maka akan semakin tepat dokumen dikategorikan.

Distribusi *f-measure* dari dataset JawaPos dapat dilihat pada gambar 5. Dapat dilihat dari gambar 5 bahwa ketepatan dokumen berita diklasifikasikan berdasarkan tiga kelas diatas cenderung merata. Nilai *F-measure* terbesar berada pada kategori sport yaitu 0,9358. Hal ini menjelaskan bahwa presentase dokumen yang diklasifikasikan dengan benar diantara semua dokumen yang termasuk kategori sport cukup tinggi dan diantara semua dokumen yang ditentukan kategorinya oleh classifier cukup tinggi pula. Namun nilai f-measure dengan metoda TFIDF dan dibersihkan noisenya dengan human analisis khususnya pada kategori politik dan ekonomi bisnis tidak cukup baik jika dibandingkan dengan metoda yang lain.

Distribusi *F-measure* dari dataset Liputan6 dapat dilihat pada gambar 6. Di sini dapat dilihat bahwa persebaran nilai *f-measure* untuk kategori politik untuk metoda TFIDF dengan human clean dan dataset before clean cenderung menurun berdasarkan penambahan jumlah atribut. Sebaliknya pada metoda *feature weighting* nilai *f-measure* untuk kategori politik meningkat. Penurunan nilai pada kategori politik untuk metoda TFIDF berdampak sebaliknya pada kategori olahraga dimana memiliki nilai yang paling tinggi yaitu 1 begitu juga untuk dataset before clean kategori olahraga memiliki nilai paling tinggi dibandingkan dengan kategori yang lain. Naik turunya nilai *f-measure* pada dataset diatas dipengaruhi dari masing-masing kategori.

Nilai *f-measure* dengan metoda TFIDF dengan dataset yang telah dihilangkan noisenya secara manual pada kategori hukum dan olahraga umumnya memiliki nilai terbesar dibandingkan dengan yang lain. Hal ini sebanding dengan nilai hasil akurasi yang diperoleh dari proses klasifikasi.

Distribusi *F-measure* dari dataset Pikiran Rakyat dapat dilihat pada gambar 7. Dapat dilihat pada gambar 7 distribusi *f-measure* untuk kategori ekonomi, nasional dan olahraga nilai tertinggi berkisar antara 0,9 sampai 1,0 pada metoda TFIDF dengan penghilangan *noise* secara manual. Persebaran nilai *f-measure* terjadi secara merata pada metoda *feature weighting*. Sedangkan nilai terendah *f-measure* ada pada dataset yang mengandung *noise*. Nilai terendah dan tertinggi ada pada kategori olahraga yaitu sebesar 0,1816 pada dataset yang mengandung *noise* dan nilai tertinggi 1 pada dataset yang telah dibersihkan noisenya dengan human analisis.

SIMPULAN

Proses deteksi *noise* pada data yang dilakukan secara manual analisis lebih baik dari pada yang dilakukan sistem. Teknik *feature weighting* untuk membersihkan blok *noise* pada halaman website dapat meningkatkan hasil klasifikasi pada datamining berdasarkan hasil ujicoba perhitungan *f-measure* pada keempat dataset. Metoda pembobotan TFIDF 5,5%-7,5% lebih baik dari pada pembobotan *feature weighting* jika menggunakan dataset yang telah terbebas dari *noise* berdasarkan hasil uji dari keempat dataset yang dihilangkan noisenya. Teknik ini berhasil membersihkan blok *noise* dan meningkatkan hasil klasifikasi berdasarkan uji coba pada 4 dataset.

DAFTAR PUSTAKA

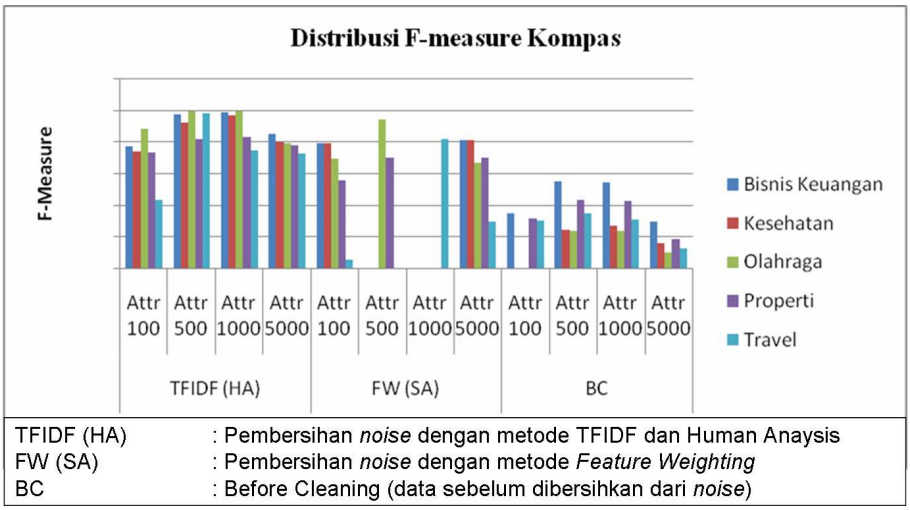
[1] Bar-Yossef, Z., Rajagopalan, S.: *Template Detection via Data Mining and its Applications*. In: Proceeding of the 11th International Conference on WWW, New York, USA. (2002)

[2] Beeferman, D., Berger, A., Lafferty, J.: *Statistical Models for Text Segmentation*. In: Machine Learning. Volume 34., Spriger, Pittsburgh, USA (1999)

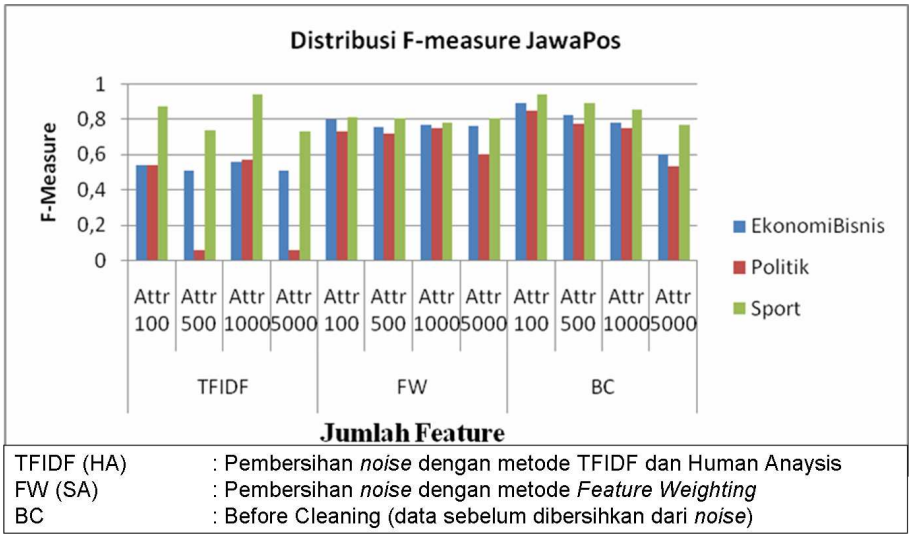
[3] Vieira, K., da Silva, A.S., Pinto, N.: *A Fast and Robust Method for Web Page Template Detection and Removal*. In: Proceeding of the 15th ACM International Conference on Information and Knowledge Management, Virginia, USA (2006)

[4] Lan Yi, Bing Liu: *Web Page Cleaning for Web Mining through Feature Weighting*. In: Proceedings of Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03), Acapulco, Mexico (August 2003)

[5] Lan Yi, Bing Liu, Xiaoli Li: *Eliminating Noisy Information in Web Pages for Data Mining*. In: Proceedings of the ACM SIGKDD International Confer-



Gambar 4: *F-measure* kompas sebelum dan sesudah deteksi *noise*



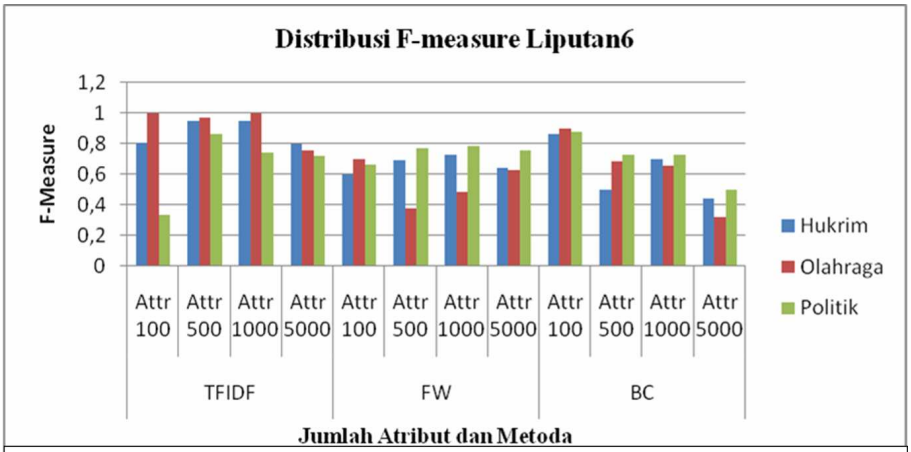
Gambar 5: *F-measure* jawapos sebelum dan sesudah deteksi *noise*

ence on Knowledge Discovery & Data Mining (KDD-2003), Washington DC, USA (August 2003)

[6] Lin, S.H., Ho, J.M.: *Discovering Informative Content Blocks from Web Documents*. In: KDD-02, Proceeding of the Eight ACM SIGKDD international Conference on Knowledge Discovery and Data Mining, New York USA (2002)

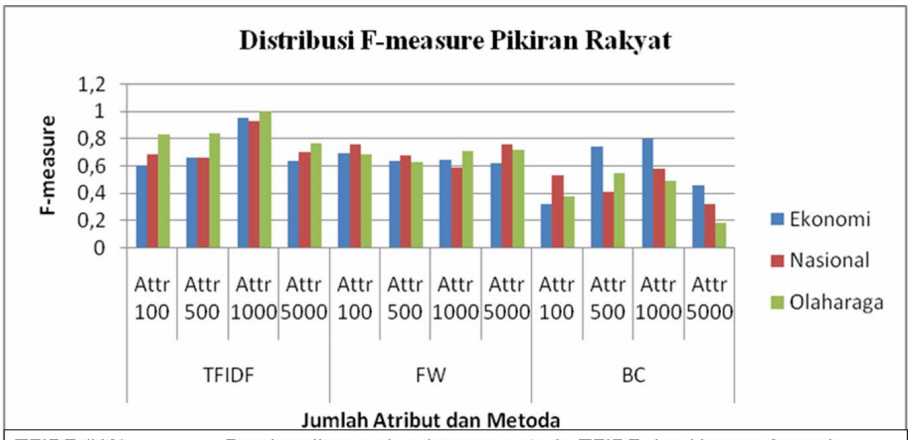
[7] Ma, L., Goharian, N., Chowdhury, A.: *Extracting Unstructured Data from Template Generated Web Documents*. In: Proceeding of the twelfth International Conference on Information and Knowledge Management, New York USA (2003)

[8] Forum, W.: *HTML DOM Tutorial*. `http://www.w3schools.com/htmldom/default.asp` 3 February 2008.



TFIDF (HA)	: Pembersihan *noise* dengan metode TFIDF dan Human Anaysis
FW (SA)	: Pembersihan *noise* dengan metode *Feature Weighting*
BC	: Before Cleaning (data sebelum dibersihkan dari *noise*)

Gambar 6: *F-measure* liputan6 sebelum dan sesudah deteksi *noise*



TFIDF (HA)	: Pembersihan *noise* dengan metode TFIDF dan Human Anaysis
FW (SA)	: Pembersihan *noise* dengan metode *Feature Weighting*
BC	: Before Cleaning (data sebelum dibersihkan dari *noise*)

Gambar 7: *F-measure* pikiran rakyat sebelum dan sesudah deteksi *noise*